Directional Hearing Aid

David Luong and Mark Piper

Advised by Professors Erik Cheever and Carr Everbach
Swarthmore College
Engineering 90 Senior Design Project
May 4th, 2006

TABLE OF CONTENTS

ABSTRACT

This report describes the work of Swarthmore students Mark Piper and David Luong in Spring 2006 and their efforts to design and construct a directional hearing aid under the guidance of Professor Erik Cheever and E. Carr Everbach. The content of the report discusses the formulation of the theory of microphone arrays in achieving directional hearing and documents the progress made in hardware and software. The system was built with four equally spaced microphones in an array connected to a small amplifier board. One of the primary goals is to demonstrate directionality in hearing aids and implement the system in a digital environment. A successful characterization of directionality has been done in theory and software and allows for design exploration of microphone arrays. A fully successful testing instrumentation has yet to be implemented, but much progress has been made in identifying current complications to allow for future work.

INTRODUCTION

We are quite fortunate to live in a world rife with all variety of rich, complex sound. Inspiring symphonies, screeching blackboards, whistling bluebirds, droning machinery, and chatting friends all fall within the plethora of what is audible. The problem arises because people need to make sense of it all simultaneously. Aural cognition plays a substantial role in the ability of an individual to function and communicate effectively in modern society. Unfortunately, the general process of aging naturally works to impair this incredibly useful ability, not to mention many other possible causes for hearing difficulty. The hope is that through clever engineering design we may help overcome these barriers of auditory function.

The concept of an electronic hearing aid is not novel, and in fact, many products already exist in the market targeted at hearing impaired population. Generally, these are small in-ear devices that amplify sound, allowing a wearer to notice sounds that would otherwise have been inaudible. These products are very well established as consumer goods and are used frequently, particularly by the elderly.

A problem with the standard design is that it does very little to distinguish between desired and undesired sound. Thus, the user is still left on her own to sift out the noise from what she wants to listen to, and this is not a trivial task. A useful modification would be to have a "smart" hearing aid that could diminish the amount of unwanted noise in comparison to the desired sound.

Our goal here is to improve upon the basic design by adding a degree of directionality as an improvement to traditional hearing aids. The device will be "smart" in that it will preferentially amplify a direction of interest compared to other directions, effectively focusing the attention of the user and diminishing background noise. This can be accomplished by branching out from an in-ear device with a single microphone to a wearable array of microphones.

Previous work has been done at Swarthmore College on such a project. In 2003, Emily Eddy developed an analog version of such a device for her senior design project. Though

successful, the project was limited by its analog nature. We seek to digitize the design. The most substantial improvement this offers is the possibility of scaling down the device size to make for a practical consumer product. The digital world also allows for a degree of consistency and durability that analog lacks. Also time delays—essential to the operation of directional hearing aids—are especially difficult to implement in analog form. Thus digital seems the only natural choice for our design.

Goals

The primary goals of this project are
1) to fundamentally understand the acoustics and physics behind directional hearing technology
2) to digitally implement a directional hearing aid in hardware and software, and
3) to instructively demonstrate the degree of directionality attained with processed measurements.

SYSTEM DESIGN OVERVIEW

The purpose of this report is to guide the reader through the development stages of understanding and implementing a directional hearing aid.

A microphone array with known separation between each microphone allows for known sound delays. A geometrical analysis of the array provides a mathematical determination of these delays. With the idea of constructive and destructive addition of physical signals in hand, we report a methodology to achieve directionality first with a 2 microphone array and then show the improvements with 4 microphones.

For physical implementation of the system, we adapt the theoretical results into an equivalent MATLAB algorithm to process the microphone signals digitally. By generating static sinusoidal signals in place of actual signals and feeding them to the algorithm, we can create simulated results demonstrating the degree of directionality achieved.

Developing the hardware of the system allows for static testing of the algorithm on actual signals and real-time testing on a digital signal processor. Selection and acquisition of the microphones dictate the amplifier design constraints. Once tested and manufactured as a printed circuit board, the amplifier attaches to the chosen end-fire microphone array ready for directionality measurements.

The Swarthmore College Sound Booth Lab is equipped with a data acquisition system and an anechoic chamber which provided the location to gather sound measurements. Further MATLAB scripts were written for efficient data acquisition and signal processing.

A graphical outline of the design process is shown in Figure 1.

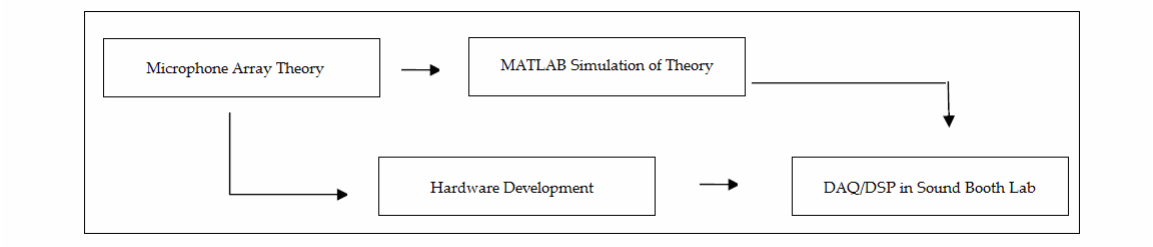


Figure 1: Flow chart outlining system design.

THEORY OF MICROPHONE ARRAYS AND ACHIEVING DIRECTIONALITY

In recent years, microphone array technology has emerged as a reliable method to improve performance in hearing aids. Traditional hearing aids rely on single microphones placed strategically on a user that amplifies a sound signal that is then played back to the user. Unfortunately, noise—signal and undesired sound—is amplified along with the signal and tends to produce an unintelligible overall signal and the user finds difficulty understanding the amplified sound.

A multi-microphone system set up as an array with a specified distance between each microphone offers a solution that can boost intelligibility of sound signals through directionality. Essentially, this is accomplished by amplifying sound arriving from a direction of interest relatively more than undesired sources from other directions.

Achieving Directionality with a 2 Microphone Array

Consider the simple case of a two microphone end-fire array as shown in Figure 2.

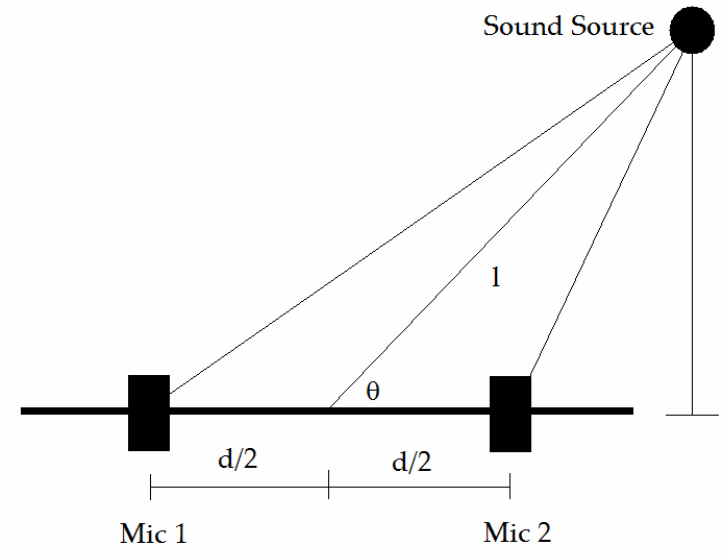


Figure 2: End-fire microphone array arrangement.

With the sound source a distance *l* away from the center of the array and the microphones placed a distance *d* apart, the two signals seen by the microphones will have an inherent delay in time. In representing the delay, we assume the sound source radiates spherical waves at c = 345 meters per second. In a geometrical analysis, we calculate the distance sound must travel to each microphone and divide it by the speed it traveled to acquire the travel time. We have

$$delay_1 = \frac{\sqrt{(l\cos\theta + \frac{d}{2})^2 + (l\sin\theta)^2}}{c}$$

$$delay_2 = \frac{\sqrt{(l\cos\theta - \frac{d}{2})^2 + (l\sin\theta)^2}}{c}$$

Representing the sound signal as a sinusoid of amplitude A with a particular frequency f, we can represent the physically delayed microphone signals by

$$output_{mic_1} = A\sin(2\pi f(t - delay_1))$$
$$output_{mic_2} = A\sin(2\pi f(t - delay_2))$$

Note that the delays are functions of theta, indicating that different sound source positions produce sinusoids received by the microphones with different delays.

Up to this point we have focused on representing the physical nature of sound. We are now ready to impose directionality. Let's consider leftward directionality, meaning we desire more amplification in the 180 degree position (in polar coordinates) and relatively less at other positions. Imagine a sound source from the right side (at 0 degrees). A sound will arrive at microphone 2 before microphone 1 due to the inherent delay from the spacing. Choosing to playback microphone 2 in real-time, we artificially impose additional delay to microphone 1 such that it adds destructively to microphone 2, resulting in an output signal with less amplitude around zero.

For leftward directionality, we want to eliminate rightward sounds as best possible. At the extreme we desire a null at the 0 degrees position. This is achieved when the two signals are half a period out of phase. By imposing a delay of *d*/c—based on the microphone separation distance—on microphone 1, the signals add destructively at a frequency where the signals are exactly time-shifted by half a period. For other frequencies and angles the signals will have partial destructive addition because they are not exactly out of phase by the proper amount. However, leftward signals will add constructively, creating leftward directionality.

To measure the directionality, we determine the maximum amplitude of the output signal at each angle. We define the output signal as

$$output = A\sin(\omega(t - delay_1 - delay_{imposed})) + A\sin(\omega(t - delay_2))$$

where ω is $2\pi f$. The maximum amplitude of this signal is given by

$$\left| 2A\cos(\omega \frac{-delay_2 - (-delay_1 - delay_{imposed})}{2}) \right|$$

More explicitly, it is given by

$$\left| 2A\cos(\omega \frac{\sqrt{(l\cos\theta + \frac{d}{2})^2 + (l\sin\theta)^2} - \sqrt{(l\cos\theta - \frac{d}{2})^2 + (l\sin\theta)^2} + d}{2c}) \right|$$

Refer to Appendix A for the derivation. Plotting on a polar coordinate system for specified parameter values given in Table 1, we generate a beam profile of the hearing sensitivities as a function of angle. This is shown in Figure 3 below.

Microphone spacing (d)	Source distance from array (l)	Microphone Amplitude (A)	Speed of sound in air (c)	Frequency (f)
.05 meters	1 meter	1 volt	345 m/s	1750 Hz

Table 1: Array Parameters

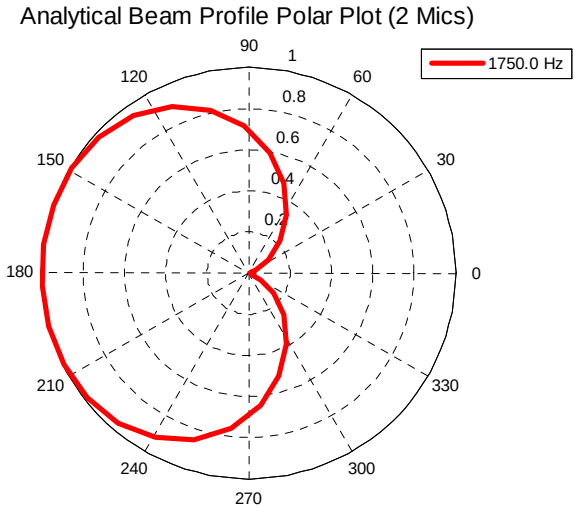


Figure 3: Leftward Directional Beam Profile for 2 Microphone Array

Note that the output signals are scaled by the number of microphones producing a magnitude between 0 and 1. We observe the nullification from the right side (0 degrees) at this particular frequency where the microphone signals add destructively. At the opposite extreme is constructive signal addition resulting in maximum hearing sensitivity. The angles in between results in sensitivities between the maximum and

minimum as expected. The resulting beam profile is an improvement through directionality by focusing on sound sources from the left and less so in other directions.

Achieving Directionality with a 4 Microphone Array

Similar derivations can be made for other sized microphone arrays. Applying the same directionality scheme for the four microphone array (refer to Appendix A for derivations), the resulting beam profile with the same specifications in Table 1 is shown in Figure 4.

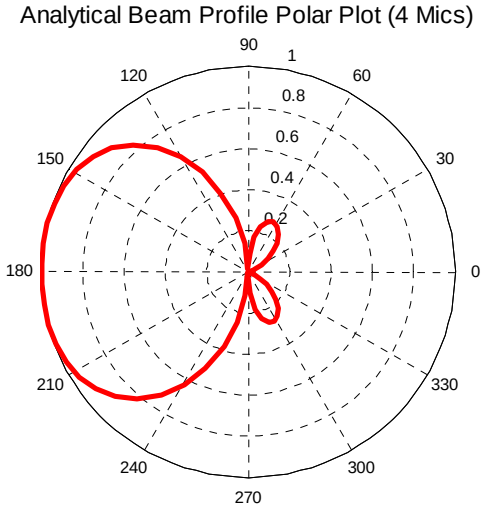
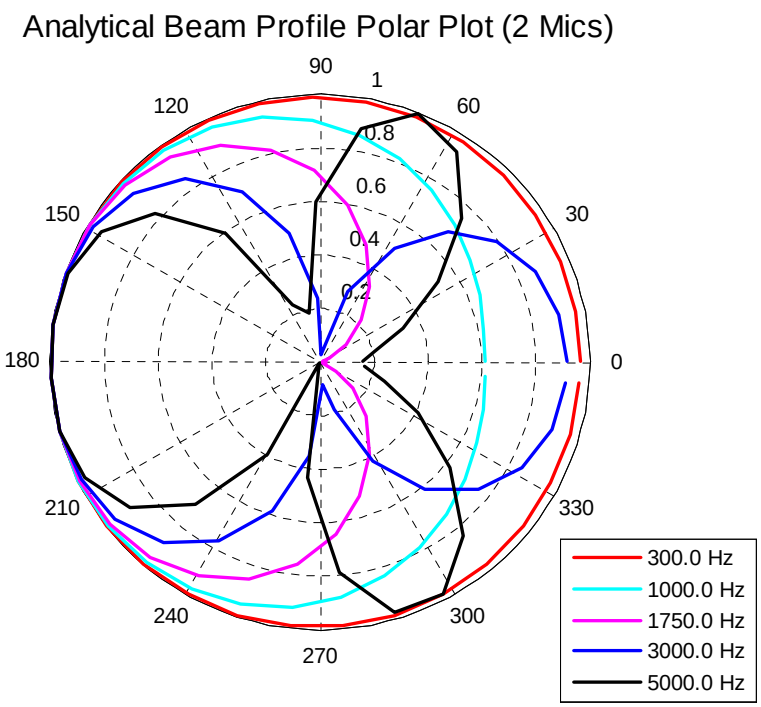


Figure 4: Leftward Directional Beam Profile for 4 Microphone Array

Comparing 2 and 4 Microphone Arrays

Comparing Figure 3 to Figure 4, we notice two significant differences. First, the main lobe appears narrower for the 4 microphone case. In focusing on the direction of a sound source, this is seen as an improvement. The second point is the appearance of side lobes, seen here at approximately 60 and 300 degrees. This can be explained by the nature of summing signals not completely in-phase or out of phase, resulting in residual signal amplitude.

In a hearing aid application, we are interested in examining these beam profiles within the range of human speech. This applicable range is between 600 Hertz to 4000 Hertz, with 3000 Hertz being most sensitive. Figure 5 examines beam profiles for several frequencies for the 2 and 4 microphone cases.



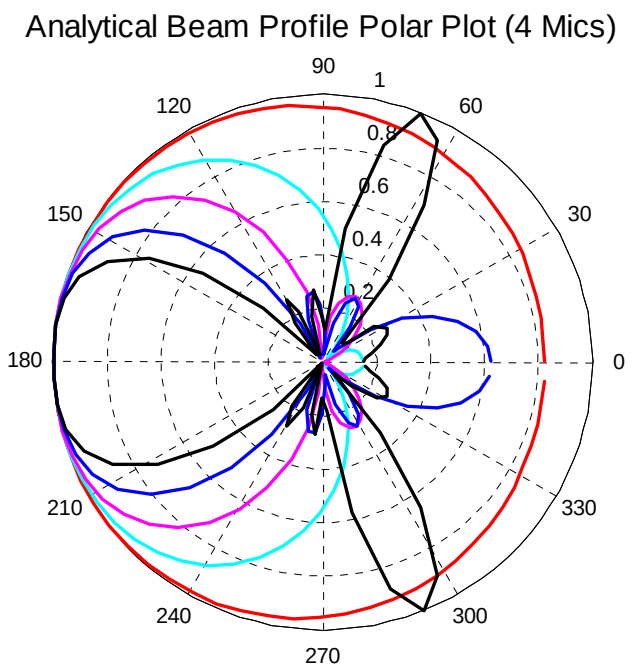


Figure 5: Directional Beam Profiles for Human Speech

The matter of choosing 2 or 4 microphone arrays depends on the tradeoff between narrower main lobes and existence of side lobes. A way to resolve this issue is by quantifying the amount of directionality of beam profiles plot. We adopt the ratio of maximum amplitude to the average amplitude as our measure of directionality. Table 2 shows these measures for the frequencies in Figure 5 for the 2 and 4 microphone arrays.

Frequency (Hertz)	300	1000	1750	3000	5000
2 Microphone	1	1.2	1.6	1.4	1.5
4 Microphone	1.1	1.6	2	2	2

Table 2: Directionality Numbers

In both arrays, we see a trend of increasing directionality for higher frequencies. This is expected since lower frequencies are less directional since their wavelengths are larger than the length of the array. For the 2 microphone case, we see the beam profile most directional at 1750 Hertz. We observe an improvement at the same frequency from the 4 microphone array due to the narrower main lobes. Furthermore, we observe little, if any, change in the directionality numbers at higher frequencies at the expense of more side lobes.

Given the theoretical implications of microphone arrays, our design choice is influenced by the following factors. With the analytical solution implemented in Matlab, different parameters were tested to yield a highly directional system within the human speech frequency band (600-4000 Hz) that would also be feasible as a device worn by a user. The 4 microphone array with specifications given in Table 1 provided a compromise between array size, directionality numbers in the frequency range, and side lobe generation.

HARDWARE DESIGN

Microphone Amplifier

The choice of microphones was a recommendation from Knowles Electronics. With an "invisible" system in mind, the EK-3024 microphones were small units capable of attaching to jewelry, eye glasses, and other accessories. They also provided ultra low noise operation for cleaner signals and were readily available in from vendors.

Testing of Knowles EK-3024 microphones was needed to "clean" their signals before data processing. Because the raw microphone signal's gain was low (in the range of mV) and noisy, we built filters and gain elements using TLC2772C Rail-to-Rail operational amplifiers. Single supply design was chosen in anticipation of battery pack inclusion in

the final system, and will be discussed in more detail later. Figure 6 shows a circuit block diagram and Figure 8 the schematic of the microphone amplifier. Note that the same design is used four times to allow feeds from our four input channels. The circuit was turn-keyed into an Ultiboard file for printed circuit board production.

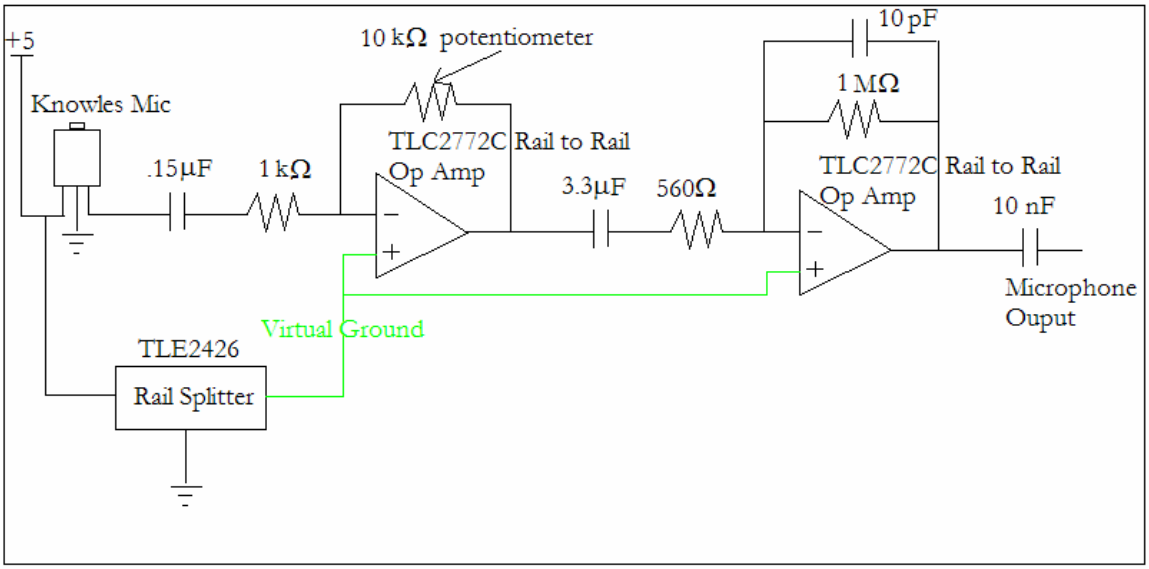


Figure 6: Microphone Circuitry

Single Supply Design

Anticipating portable battery power in the system, we considered single supply design. A rail-splitter effectively creates a virtual ground, specified at $V_{cc}/2$ or 2.5 volts with a 5 volt supply. Connecting this to the positive terminals of the operational amplifier references the microphone signal to the virtual ground. With a capacitor to remove the DC component, the result is a signal range between -2.5 and 2.5 volts relative to virtual ground.

Amplification and Gain Stages

The gain and filters were divided into two stages to avoid op-amp saturation. The gain was experimentally chosen to achieve sufficient dynamic range on the oscilloscope; a 10 kOhm potentiometer later replaced the feedback resistor in the first stage to allow for adjustable volume control. By including the potentiometer, we handle possible signal clipping, which depends on the amplifier gain as well as varying source amplitude at different frequencies.

Before processing, the signal was cleaned as best possible to reduce processing errors and recreate the signal digitally. We used a simple analog band-pass filter to remove undesired low and high frequency content inherent in the microphone signals. With the available values of resistors and capacitors in the lab, the frequency band turned out to be in the range between 100 Hz and 10 kHz, inclusive of human speech and hearing.

It was later discovered that the acquired signal possessed a floating ground when attached to a low impedance DAQ board. Attaching a 1 Mohm resistor from the amplifier output to ground solves the problem without significant effect on the output amplitude.

Because we use four microphones, we have replicated the amplifier above for each. The final amplifier board is shown in Figure 7 below along with its circuit layout in Figure 8.

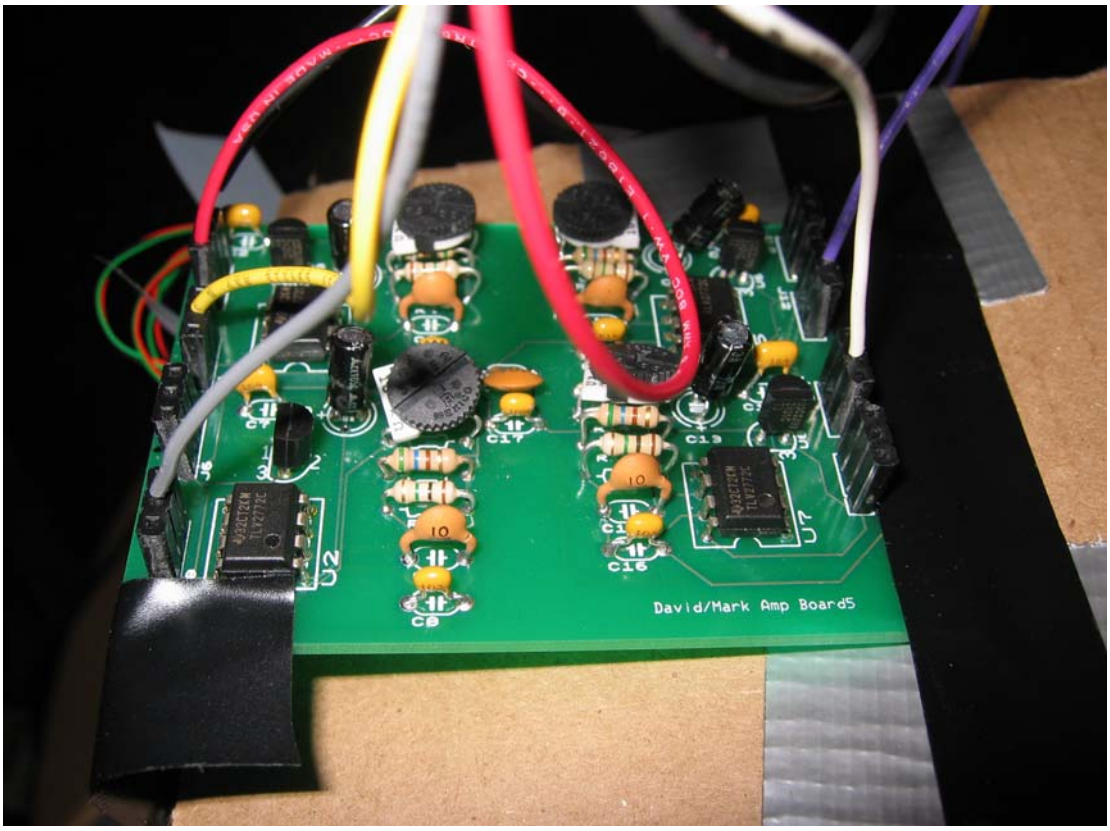


Figure 7: Amplification Board

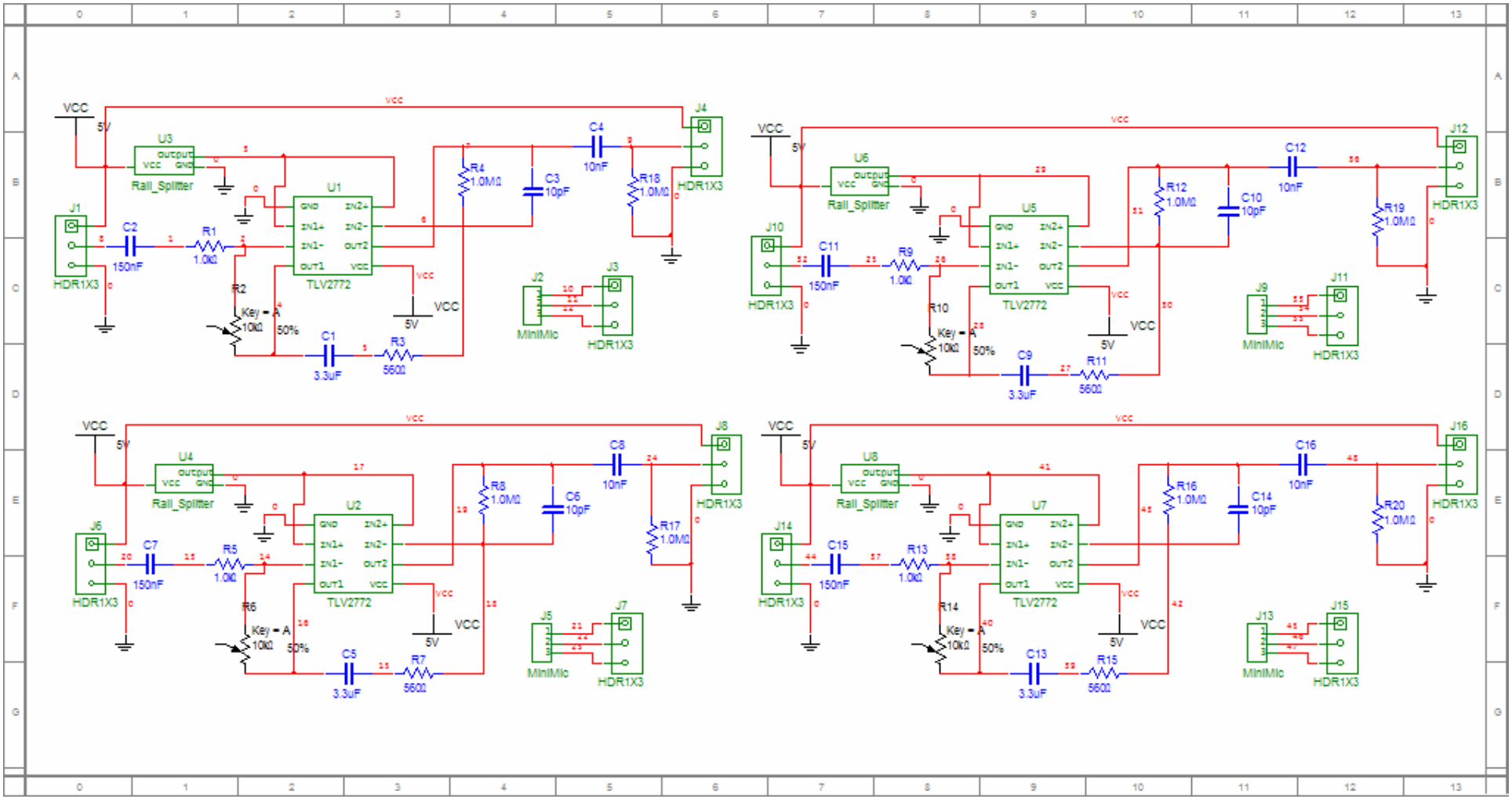


Figure 8: Multisim Microphone Amplification Layou

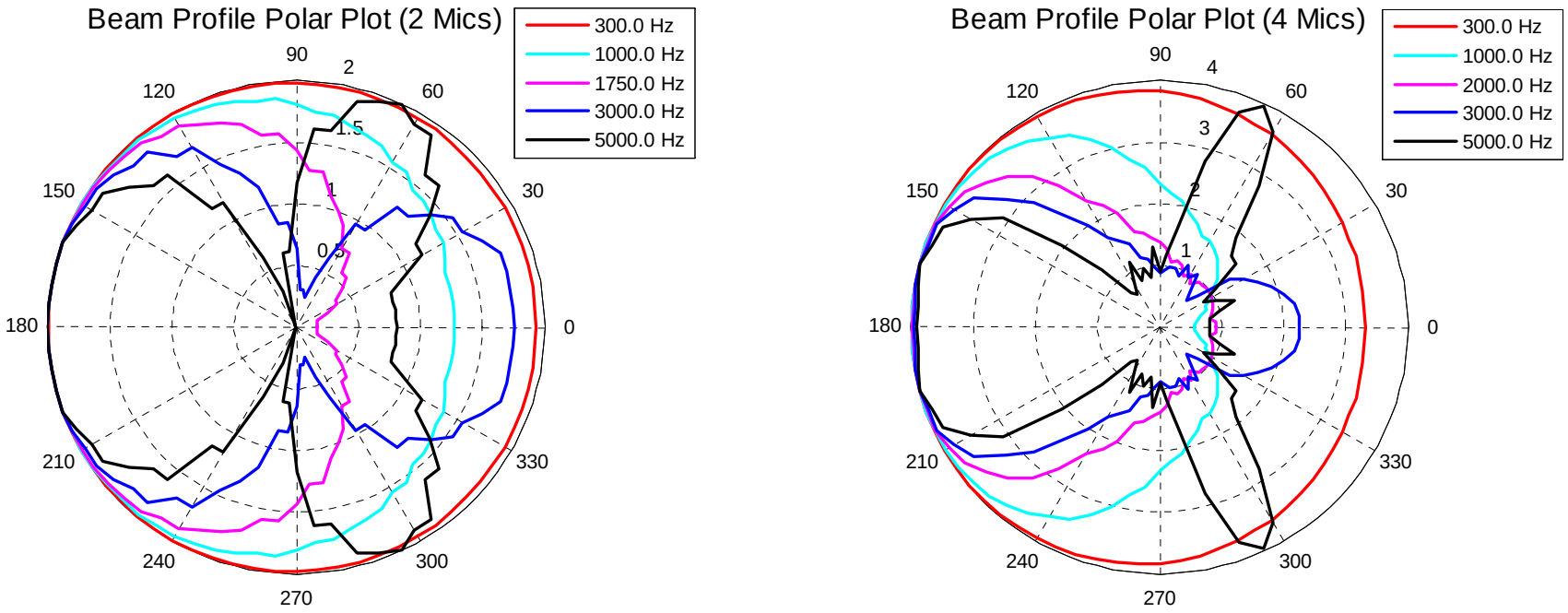
SOFTWARE DESIGN

In addition to our fully analytical theoretical analysis, we also performed a MATLAB simulation of both two and four microphone arrays. We define the sound source as a simple sinusoidal of amplitude one. To determine the signal heard at each microphone from such a source, we used simple trigonometry to solve for the distance between source and microphone. This translates directly into the necessary delay to apply to the source when divided by the speed of sound (345 m/s).

To apply a leftward sense of directionality to our simulated microphone signals, the left microphones must be delayed with respect to the right microphones. Effectively, the goal is to be using the real-time value of the rightmost microphone while delaying each subsequent microphone to the left by an additional amount. Specifically, the amount is equal to the amount of time needed for sound to travel between any two of the evenly spaced microphones. However, since this is being done digitally, there is no easy way to apply an exact time delay to each microphone signal. Rather, it must be done in terms of an array index delay. Thus, each signal is delayed by an appropriate number of array indices.

Once the proper delays have been applied, the new signals can all be summed together. Sound sources coinciding with the sense of directionality tend to add constructively while those far away from the sense of directionality will add destructively. Because we are working with simple sinusoids, a good measure of the amount of directionality is simply the maximum magnitude of the resultant summed periodic signal. The higher the degree of constructive addition the higher the maximum will be and the more destructive addition the lower the maximum will be.

Two loops were implemented to automate the procedure of beam profile generation. One rotates the sound source an entire 360° around the array. The second changes the frequency of the sound source. All the while, the data is being stored so that at the end of its iterations the beam profile can be plotted. The results are shown on the next page in Figure 9.



Figures 9: MATLAB simulated beam profiles for 2 and 4 microphone array.

The 2 and 4 microphone cases simulated results match well with the analytical results arrived at previously. The plots are more piece-wise, arising from the fact the results are discrete and not continuous. The trends and analysis presented earlier still apply in simulation.

TESTING SETUP AND METHODOLOGIES

With the amplifier and microphone array designed and assembled, we next made steps to verify the system's directionality performance with real sound data. We proceed to explain these steps with the following block diagram in Figure 10.

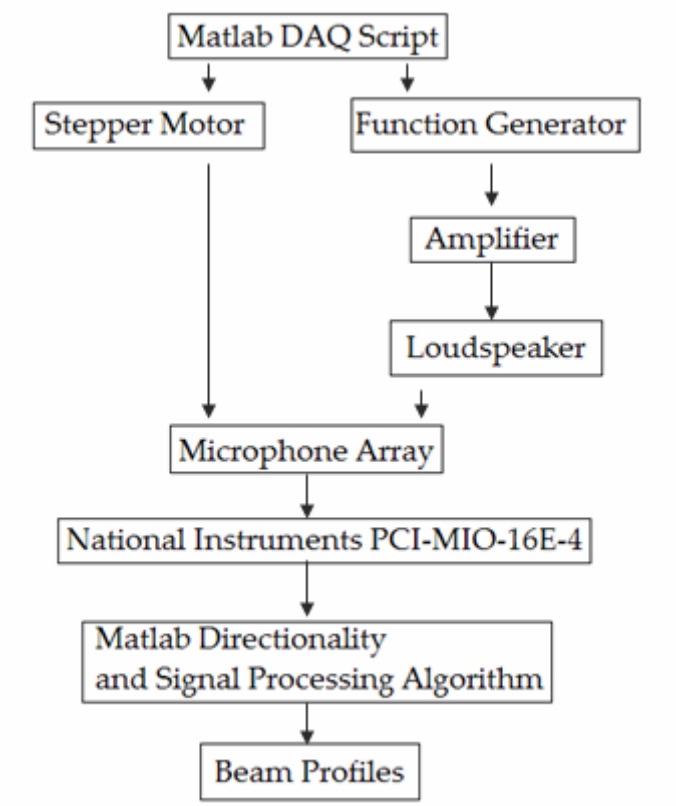


Figure 10: Testing the System

Sound Booth Lab

Measurements were taken in the Sound Booth Lab located near 3 and 5 Whittier Place of Swarthmore College. The anechoic chamber provides the most ideal environment to perform acoustic measurements with noise and echo mitigated (see Figure 11). Reverberant signals are typically a small fraction of the incident sound, and the anechoic chamber further mitigates effects and presence of standing waves from the microphone output for purposes of testing.

Data Acquisition System

A National Instruments 50 pin PCI-MIO-16E-4 board and GPI card were used in conjunction with Matlab to bring measurements into a computer for processing. The board allows for the required four channels and sufficiently high sampling rates, in our case at least 40 kHz per channel. For purposes of testing, we used 100 kHz to accurately digitize the physical signals. The board also provides two digital outputs needed to control a stepper motor used to facilitate measurements at various angles.

Several Matlab scripts were written to perform the processes during the data acquisition. Table 3 lists each script and a brief description of its functionality, and the full code can be found in Appendices D and E.

stepper_daq.m	Initializes and executes a data acquisition routine, recording measurements in a data matrix
stepper_adj.m	stepper_adj.m allows for manual adjustment of the step motor position
process_pair.m	Takes the 'data matrix', runs findmax.m on the summed signals, and returns amplitude information
process.m	Takes the 'data matrix', runs process_pair.m on even segments, and saves to a 'processed matrix' with angle and amplitude information
plott.m	Takes a 'processed matrix' and generate beam profiles
findmax.m	Algorithm to find appropriate amplitude of summed signals
setamp.m	Opens HP 3314A function generator at GPIB address 7 and sets amplitude of output signal
setfreq.m	opens HP 3314A function generator at GPIB address 7 and sets frequency of output signal

Table 3: Matlab Scripts

The motor is situated below the microphone array that rotates 96 ticks for a full revolution, allowing for 360º sound measurements at 3.75º intervals (see Figure 12). Other intervals are possible depending on the number of ticks per adjustment. Matlab scripts instruct the motor to rotate counter clockwise with a low output and clockwise for a high output.

A Hewlett Packard 3314A function generator operates from Matlab instructions and together with an amplifier, selected frequency tones are played through a JBL loudspeaker.

Software Methodology for Directionality Testing

Using the National Instruments board (see Figure 13) and the DAQ capabilities of MATLAB, we are able run the simulation described earlier with real tones. A series of tones is played with 1 second pauses between and 4 channels of data (one from each microphone) is gathered and stored in a separate array for each tone. The MATLAB script then turns the array a specified angle and the sequence of tones and data collection is repeated. This continues until a full revolution has been made.

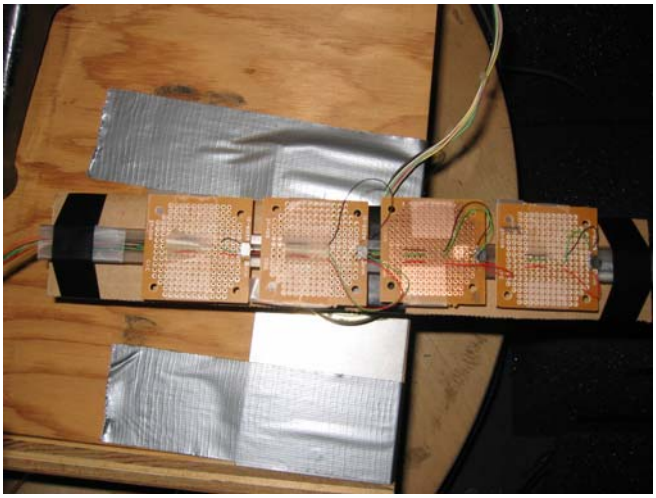


Figure 11 (left): Test setup in anechoic chamber

Figure 12 (above): 4 microphone end-fire array resting on the stepper motor

Figure 13 (below): DAQ board, amplifier, and function Generator

RESULTS AND DISCUSSION

Electrical Noise

After running our DAQ setup, we are left with a matrix containing 4 channels of microphone data for each angle at each frequency under test. Figure 14 shows a sample of one channel of raw microphone data.

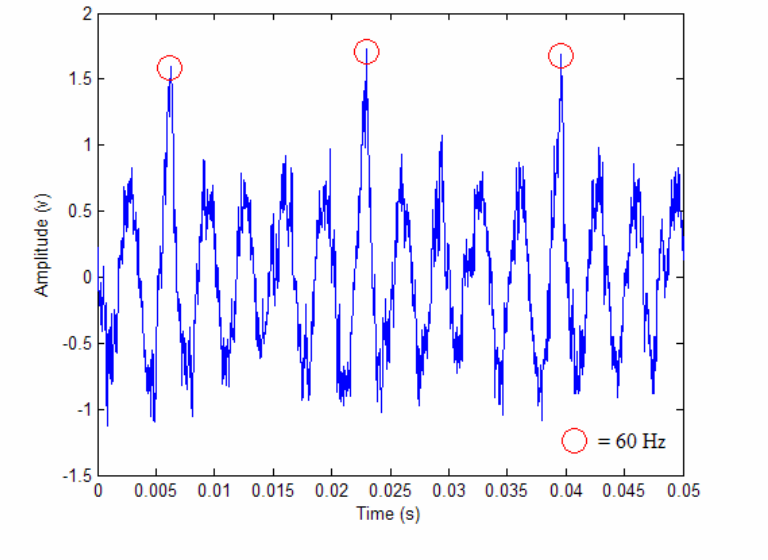


Figure 14: Raw microphone data.

The output clearly contains the expected sinusoidal component which results from the monofrequency input. However, also take notice of the peaks highlighted with red circles. These disturbances were not expected, and measurement reveals that these occur at a 60 Hz frequency. Such signal corruption is a common problem for electrical systems as this is the frequency of the alternating current utilized in the electrical grid. The presence of many instruments and equipment operating on 60 Hz power in close proximity of the wiring carrying our microphone signals during testing is a likely source of our signal interference.

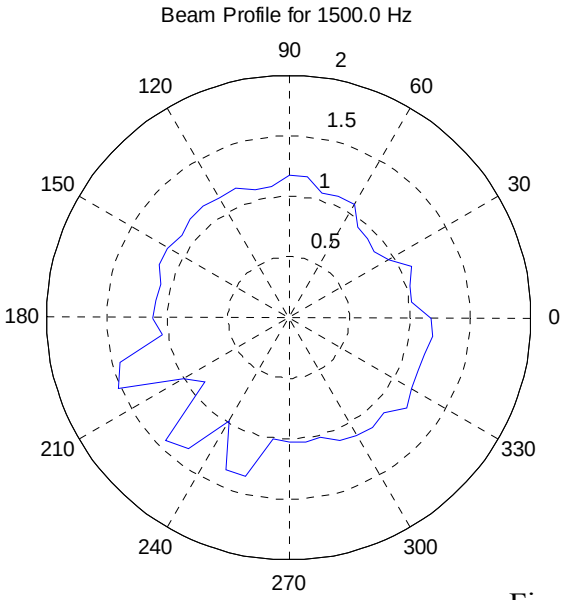
Our solution to this problem was to implement a digital high-pass filter to rid ourselves of these extra peaks. We chose to do this using a 4^{th} order Butterworth filter with a cutoff frequency of 200 Hz. The chose cutoff is high enough to significantly attenuate the 60 Hz signal while not infringing on the frequency band of human speech beginning at 400 Hz. The order of the filter did not have to be very high as the specifications of transition width and ripples on the filter did not have to be particularly stringent.

Normalization

We also normalized all of our microphone inputs in code to try and compensate for microphone mismatches. We attempted to correct for this in our analog amplifier circuitry as well by tuning potentiometers but decided also to include this extra measure in software for further accuracy.

Experimental Beam Profiles

With the microphone signals adjusted appropriately, we then applied the same algorithm to the real data that had proven effective in simulation to arrive at the beam profiles shown in Figures 15 and 16.



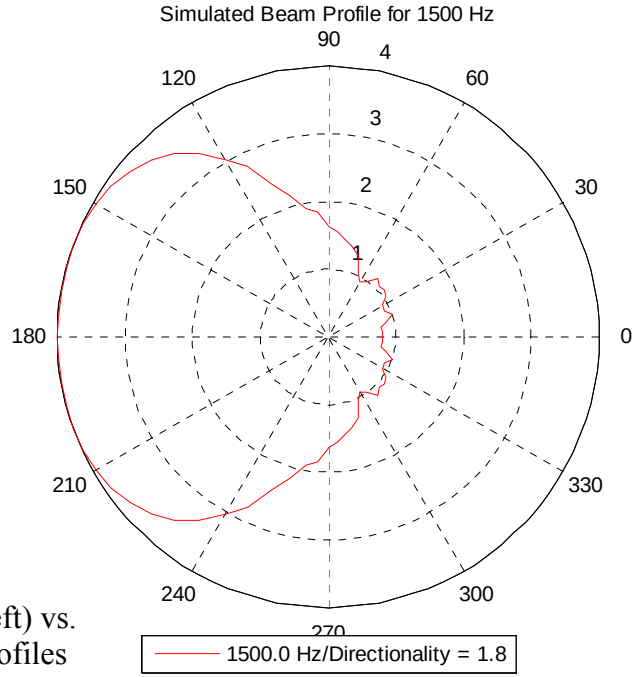
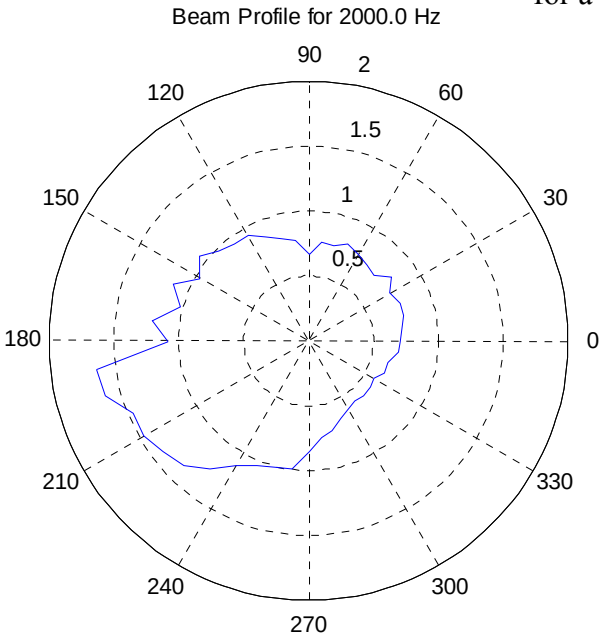
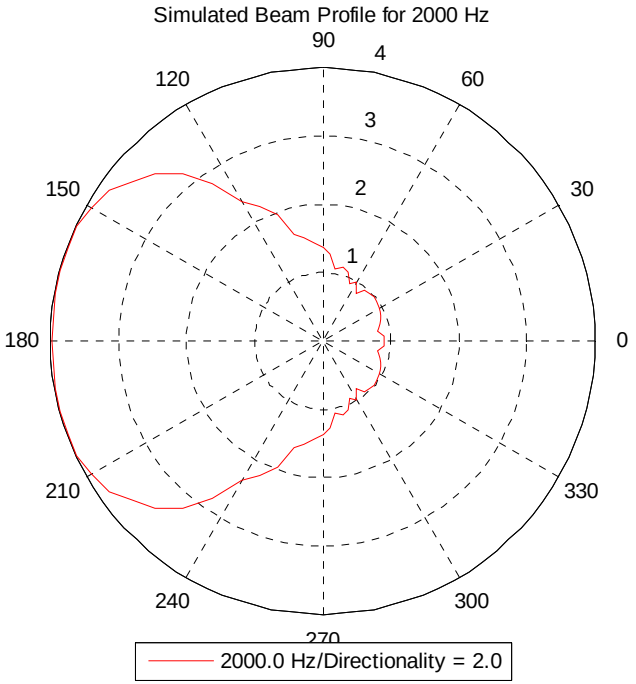
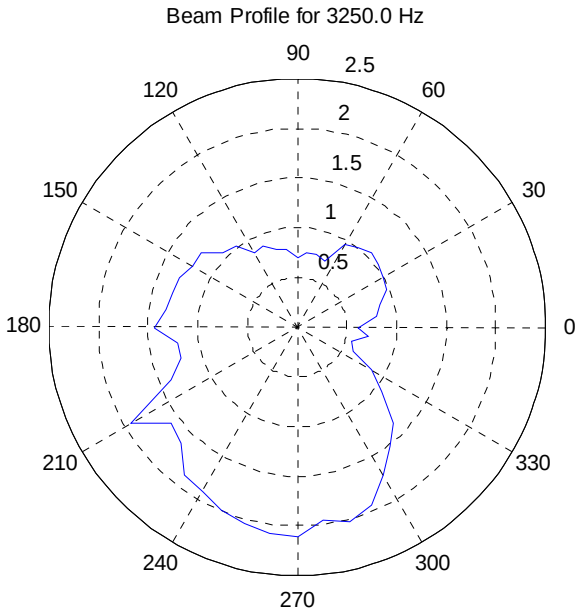


Figure 15: Experimental (Left) vs. Simulated (Right) Beam Profiles for a 4 Microphone Array at 1500 and 2000 Hz.







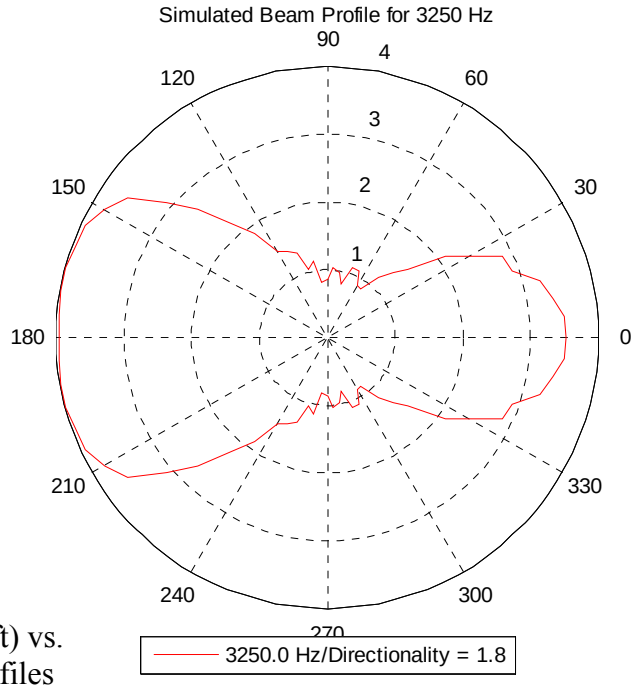
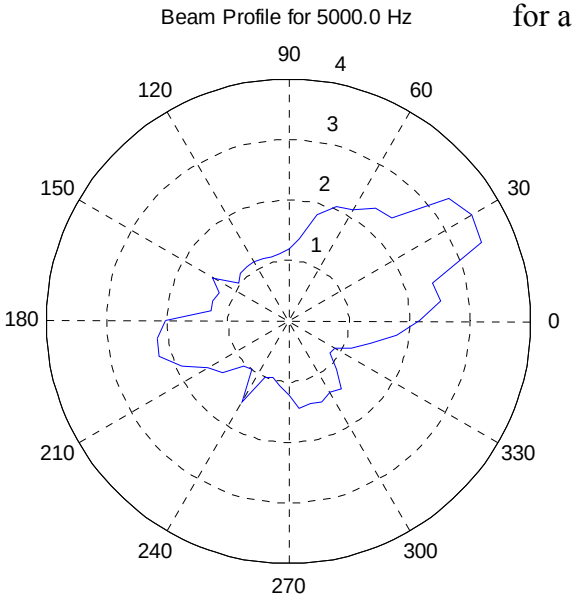
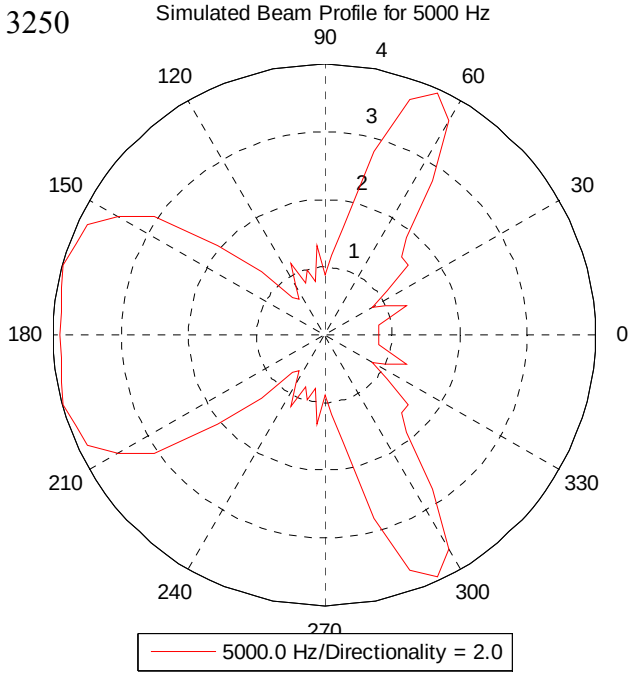


Figure 16: Experimental (Left) vs. Simulated (Right) Beam Profiles for a 4 Microphone Array at 3250 and 5000 Hz.





Unfortunately, the results do not agree with the simulated responses. Directionality is ambiguous at best in cases like at 2000 Hz and the symmetries we would expect do not appear except in the 1500 Hz case. The one general trend that does match is that there is relatively more variation in the beam profiles with higher frequency inputs which is expected because of the development of side lobes.

Loudspeaker Nonlinearities

A minor explanation for the observed discrepancies is the nonlinear behavior of the loudspeaker that we used in our data acquisition. Testing revealed that two tones relatively close in frequency and equal in amplitude played to the microphone array could potentially result in drastically different outputs. We attempted to correct for this in our procedure by choosing particular frequencies for which the signal produced by the loudspeaker was of substantial and equal magnitude. The four frequencies represented in Figures 15 and 16 were chosen in this manner. However, the processing of correcting for the variations was far from rigorous.

Signal-to-Noise Analysis

Another explanation for the results is the poor signal-to-noise ratio (SNR) of our system. Measurement showed that the SNR for our system was 10.94 dB which translates to an output where the signal is only about 3.5 times the level of the noise. Likely, some of this is due to crosstalk in the amplifier circuitry and in the wiring used to channel the data from the anechoic chamber into the data acquisition setup outside. To get a visual look at the SNR we show in Figure 17 the very end of a tone test.

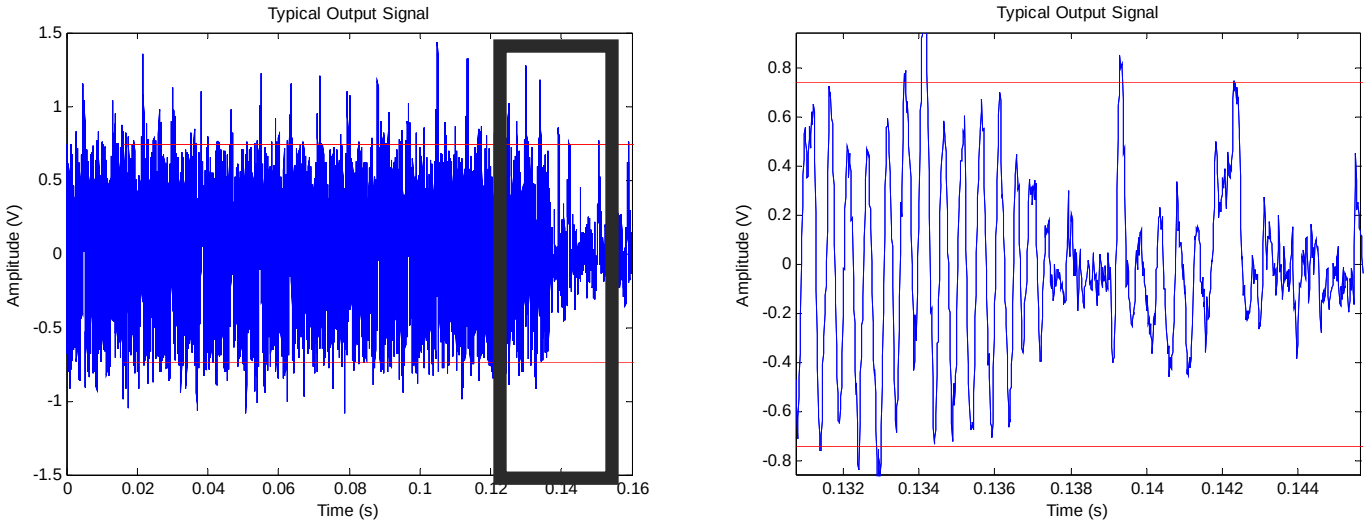


Figure 17: The end of a tone test and a magnified version of the end of the test.

Once the input halts, there is still a significant nonzero output. This can only be attributed to noise in the system, and it is on the order reported in the SNR.

Algorithmic Anomalies

Noise seemed to cause problems for the part of our algorithm for generating beam profiles from raw data which requires an estimation of the amplitude of a signal. The function *findmax.m* was written to accomplish this task (the full code can be found in Appendix E). Though it works to guess an appropriate value for the amplitude of a noisy signal a large majority of the time, a few special data sets appeared to confuse our algorithm as can be seen in Figure 18.

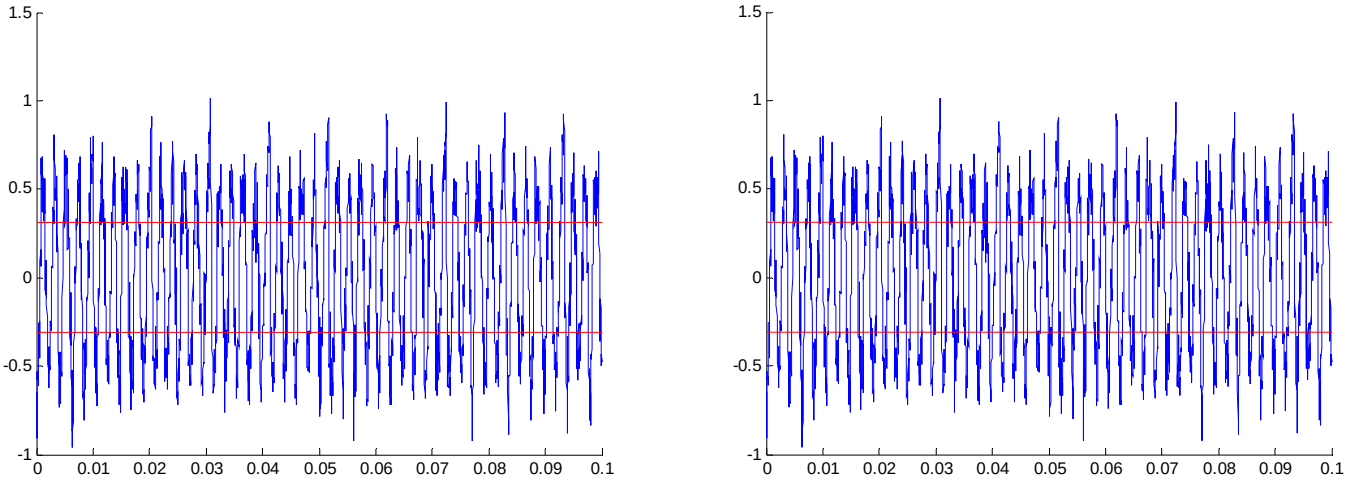


Figure 18: Two results from *findmax.m*.

The horizontal red lines represent the amplitude estimate of *findmax.m* given the signal in blue. The particular noise in these two cases seems to have cause out algorithm to fail at a few instances.

One last piece of evidence that noise is the source of fault in our system is the inconsistency in results of repeated identical tests. This is shown in the beam profiles of Figure 19.

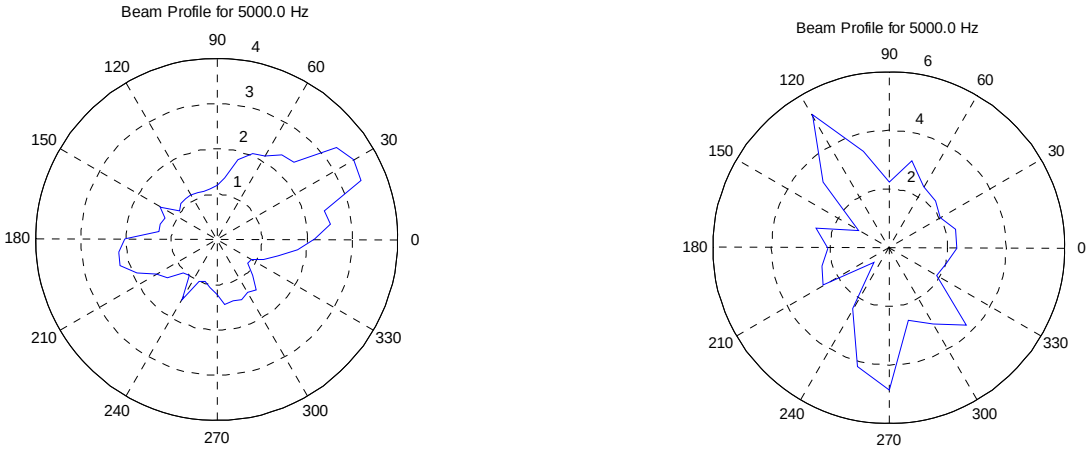


Figure 19: Beam profiles from two identical tests.

These results lack similarity and thus it is reasonable to conclude that there is some random process present to cause such discrepant results in identical tests. Noise is such a random effect and likely the culprit.

FUTURE WORK

Much progress has been made toward understanding and implementing a directional hearing aid in hardware and software. Our system has potential to lead to further development in various areas after performing instrumentation improvements.

Issues regarding noisy microphone signals need resolving prior to further developments. As pointed out earlier, the current instrumentation requires protection from environmental disturbances such as 60 Hertz electrical equipment, which is common in many systems. Once completed, the Matlab and National Instruments data acquisition protocol is already set up to robustly test directionality with ease.

The theoretical and simulated characterizations of directionality provide the structure and algorithm to model directional hearing and explore changes from varying system parameters such as microphone spacing distances, number of microphones, etc. Optimal designs can be formulated in further development. For example, steering the directionality of interest might provide an informative exploration. It has been implemented in the analytical scripts, but has yet been fully explored.

The scripts written to illustrate directionality provide a structure to readily translate the algorithm into real-time platforms where incoming sounds can be processed and played back to the user through a speaker device. Simulink, a tool branch of Matlab capable of realizing real-time developments, is a sensible direction to head in first and ultimately a DSP device a hearing impaired individual can wear. The Swarthmore engineering department has recently acquired the real-time development application in Simulink. Also, we have carefully chosen a DSP development kit by Analog Devices capable of handling four analog input channels and adequate sampling rates.

Because hearing impairments vary across individuals, testing subjects with the developed hearing aid is essential. Measures could be developed to quantify improvements in speech intelligibility and overall effectiveness of the hearing aid as a comfortable and non-stigmatizing device.

In keeping with our project vision of catering to an unnoticeable device, wireless development between the speaker and the processing unit is a recommended application in future student projects. Currently, such is done via telecoil and electromagnetic applications and could potentially interest students in those areas.

ACKNOWLEDGEMENTS

We would like to thank our advisors Professor Carr Everbach and Professor Erik Cheever for their goodwill, encouragement, and support throughout the entirety of the project. We would also like to thank Edmond Jaoudi for his prompt response to our every need. Also, many thanks to Helene Shapiro for help with our theoretical calculations. We would like to thank Richard Lu for his never-ending capability to entertain and for saving David's ears for a few more years.

APPENDIX

A. Analytical Method: Derivation

Given the equation

$$output = A\sin(\omega(t - delay_1 - delay_{imposed})) + A\sin(\omega(t - delay_2)) \tag{1.1}$$

we find the amplitude of *output* by summing the following two trigonometric identities.

$$\begin{aligned} A\sin(a+b) &= A\sin a\cos b + A\cos a\sin b \\ A\sin(a-b) &= A\sin a\cos b - A\cos a\sin b \end{aligned} \tag{1.2}$$

Hence,

$$A\sin(a+b) + A\sin(a-b) = 2A\sin a\cos b \tag{1.3}$$

Note that amplitude of (1.3) is $2A\cos b$. We rewrite equation (1.1) in the form of (1.3) by letting

$$\begin{aligned} a &= \omega t \\ b &= \frac{1}{2}[\omega(-delay_1 - delay_{imposed}) - \omega(-delay_2)] \\ &= \frac{\omega(-delay_1 - delay_{imposed} + delay_2)}{2} \end{aligned} \tag{1.4}$$

The maximum amplitude of output is then

$$\left| 2A\cos\left(\frac{\omega(-delay_1 - delay_{imposed} + delay_2)}{2}\right) \right| \tag{1.5}$$

Note that the parameters $delay_i$ are functions of position theta, and hence we can construct a polar plot as a function of angle, namely a beam profile as we have seen.

4 Microphone Array Case

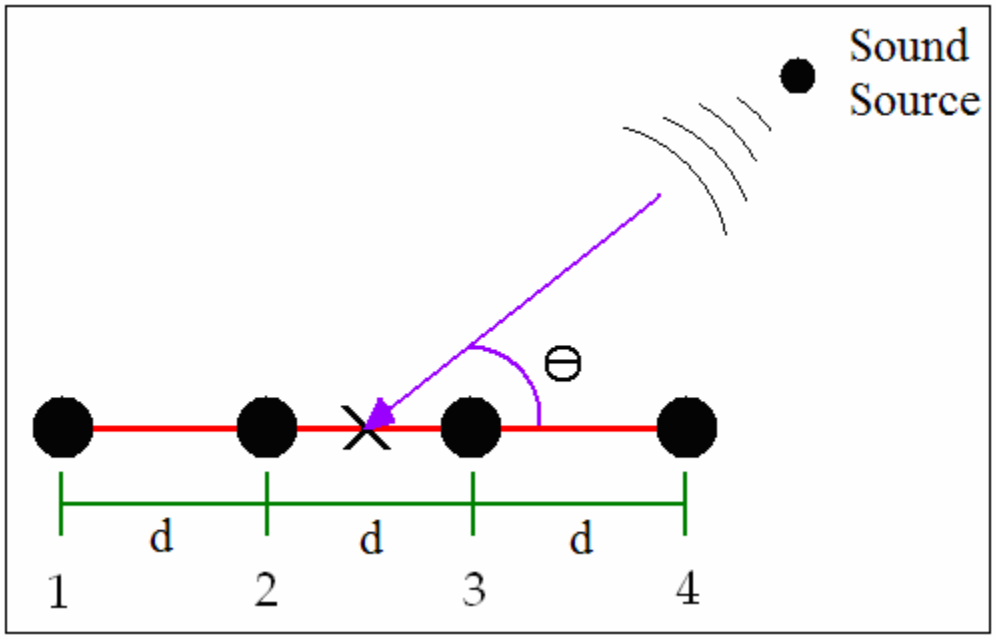


Figure 20: 4 Microphone Array

Similarly, this technique can be applied iteratively to n-microphone end-fire arrays where n is even (a separate modified technique must be used for cases where n is odd). For the case where n=4 (see Figure 20), the output signal for leftward directionality is given by

$$\begin{aligned} output &= A\sin(\omega(t-delay_1-3d))+A\sin(\omega(t-delay_2-2d)) \\ &+A\sin(\omega(t-delay_3-d))+A\sin(\omega(t-delay_4)) \end{aligned} \tag{1.6}$$

The physical delays of this microphone array can be determined from geometrical analysis and are given by the following.

$$delay_1 = \frac{\sqrt{(l\cos\theta+3\frac{d}{2})^2+(l\sin\theta)^2}}{c}$$

$$delay_2 = \frac{\sqrt{(l\cos\theta-\frac{d}{2})^2+(l\sin\theta)^2}}{c}$$

$$delay_3 = \frac{\sqrt{(l\cos\theta-\frac{d}{2})^2+(l\sin\theta)^2}}{c}$$

$$delay_4 = \frac{\sqrt{(l\cos\theta-3\frac{d}{2})^2+(l\sin\theta)^2}}{c}$$

Using (1.2) yields

$$\begin{aligned} output &= 2A\sin\omega t\cos(.5\omega(-delay_1 - 3d + delay_4)) \\ &+2A\sin(\omega(-delay_2 + delay_3 - d)) \\ &= [2A(\cos(.5\omega(-delay_1 + delay_4 - 3d)) + \cos(.5\omega(-delay_2 + delay_3 - d)))]\sin\omega t \end{aligned} \tag{1.7}$$

Thus, the maximum amplitude of the summed microphone signals is

$$\left|[2A(\cos(.5\omega(-delay_1 + delay_4 - 3d)) + \cos(.5\omega(-delay_2 + delay_3 - d)))]\right| \tag{1.8}$$

B. Analytical Method: Matlab Implementation

a. analytical2.m

```
clear all; close all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% analytical2mic.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% This is a script to produce the beam profiles of a 2 microphone
array
%% with sinusoidal inputs incident at varying angles based on
mathematical
%% derivations of the theoretical response.

%% System Parameters
L = 1; %sound source distance from mic array center
A = 1/L; %amplitude attenuation assuming spherical source
d = .05; %mic spacing [m]
c = 345; %speed of sound [m/s]

theta = 0:.1:2*pi;
f = [300 1000 1750 3000 5000];

for directionality = 1:8:31,

    for i = 1:length(f),
        w = 2*pi*f(i);

        int_delay1 = w*sqrt((L*cos(theta)+d/2).^2+(L*sin(theta)).^2)/c;
        int_delay2 = w*sqrt((L*cos(theta)-d/2).^2+(L*sin(theta)).^2)/c;

        %%steering directionality: 1 -> 32 (left @ 90 to right, CW)
        ext_delay = int_delay1(directionality)-
int_delay2(directionality);
```

```
        %delay Mic1 (left) --> left-looking directionality
        output_max(i,:) = abs(A*cos(.5*((-int_delay1-ext_delay)-(-
int_delay2))));

    end

    outLeg = '';
    color = ['k' 'r' 'c' 'm' 'b'];
    for i=1:length(f)
        curr_color = color(mod(i,5)+1);
        polar(theta,output_max(i,:),curr_color);hold on;
        outLeg = strvcat(outLeg,sprintf('%.1f Hz', f(i)));
    end
    hold off
    legend(outLeg);
    axis([-1 1 -1 1]);axis square;
    title({'Analytical Beam Profile Polar Plot (2 Mics)';''})
    dirnum = max(output_max)/mean(output_max)
    pause;

end
```

b. analytical4.m

```
clear all; close all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% analytical4mic.m %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% This is a script to produce the beam profiles of a 4 microphone
array
%% with sinusoidal inputs incident at varying angles based on
mathematical
%% derivations of the theoretical response.

%% System Parameters
L = 1; %sound source distance from mic array center
A = 1/L; %amplitude attenuation assuming spherical source
d = .05; %mic spacing [m]
c = 345; %speed of sound [m/s]

theta = 0:.1:2*pi;
f = [300 900 2300 3000 5000];

for directionality = 1:16:31,

    for i = 1:length(f),
        w = 2*pi*f(i);
```

```
        int_delay1 = w*sqrt((L*cos(theta)+3*d
/2).^2+(L*sin(theta)).^2)/c;
        int_delay2 = w*sqrt((L*cos(theta)+d/2).^2+(L*sin(theta)).^2)/c;
        int_delay3 = w*sqrt((L*cos(theta)-d/2).^2+(L*sin(theta)).^2)/c;
        int_delay4 = w*sqrt((L*cos(theta)-
3*d/2).^2+(L*sin(theta)).^2)/c;

        %no steering
        ext_delay = w*d/c;
        %Note: addition of four sinusoids matter (sum Mic1/Mic4, then
Mic2/Mic3)
        output_max(i,:) = abs(.5*A*(cos(.5*(-int_delay1+int_delay4-
3*ext_delay))+cos(.5*(-int_delay2+int_delay3-ext_delay))));

    end

    outLeg = '';
    color = ['k' 'r' 'c' 'm' 'b'];
    for i=1:length(f)
        curr_color = color(mod(i,5)+1);
        polar(theta,output_max(i,:),curr_color);hold on;
        outLeg = strvcat(outLeg,sprintf('%.1f Hz', f(i)));
    end
    hold off
    legend(outLeg);
    axis([-1 1 -1 1]);axis square;
    title({'Analytical Beam Profile Polar Plot (4 Mics)';''});
    dirnum = max(output_max)/mean(output_max)
    pause;

end
```

C. Microphone Array Simulation in Matlab

a. simulation_2mic.m

```
clc; clear all; close all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%% simulation_2mic.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% This is a script to simulate the response of a 2 microphone array to a
%% sinusoidal input incident at varying angle. The results are displayed in
%% the form of a beam profile.

%% defining the system

% 1   2     component labels
% o x o     microphones (o) and center of the array (x)

v = 345; %speed of sound in m/s
d = .05; %distance between microphones in m
l = 1; %distance of signal source from array center in m

FS = 100000; %sampling frequency
deltaT = 1/FS; %time step
n = 1000; %number of data points
t = 0:deltaT:n*deltaT;
% f = [300 1000 2000 3000 5000]; %base frequency in Hz
f = [300 1000 1750 3000 5000]
deltaTheta = 5;

%iterate over frequencies
for j=1:length(f)

    %% creation of microphone data

    %the signal
    s = sin(2*pi*f(j)*t);

    %iterate over thetas
    for theta=0:deltaTheta:360

        %receiving the signal in a microphone array
        x = l*cosd(theta);
        y = l*sind(theta);

        %calculate the delay of microphone 1
        h1 = sqrt((x+d/2)^2 + y^2);
        delay1 = h1/v;
        index_delay1 = round(delay1/deltaT);
```

```
        %calculate the delay of microphone 2
        h2 = sqrt((x-d/2)^2 + y^2);
        delay2 = h2/v;
        index_delay2 = round(delay2/deltaT);

        for i=1:n+1
            %signal from microphone 1
            if i <= index_delay1
                m1(i) = 0;
            else
                m1(i) = s(i-index_delay1);
            end

            %signal from microphone 2
            if i <= index_delay2
                m2(i) = 0;
            else
                m2(i) = s(i-index_delay2);
            end
        end

        %% applying directionality

        delay = d/v;
        index_delay = round(delay/deltaT);

        %delay signal from microphone 1
        for i=1:n+1
            if i > index_delay
                m1Delay(i) = m1(i-index_delay);
            else
                m1Delay(i) = 0;
            end
        end

        %shift signals to only look at the steady state
        ssShift = max(index_delay1,index_delay2)+index_delay;
        m1DelaySS = m1Delay(ssShift:length(m1Delay));
        m2SS = m2(ssShift:length(m2));
        tSS = t(1:length(t)-ssShift+1);

        %sum and find the amplitude of the resultant signal
        output = m1DelaySS + m2SS;
        outputMag(floor(theta/deltaTheta)+1) = max(output);

    end
    dirnum(j) = max(outputMag)/mean(outputMag);
    directionalMag(j,:) = outputMag;

end
```

```
%% plot the beam profile

figure;
outLeg = '';
color = ['k' 'r' 'c' 'm' 'b'];
for i=1:length(f)
    theta = (0:deltaTheta:360).*pi/180;
    curr_color = color(mod(i,5)+1);
    polar(theta,directionalMag(i,:),curr_color);hold on;
    outLeg = strvcat(outLeg,sprintf('%.1f Hz/Directionality = %.1f',
f(i),max(dirnum(i))));
end
hold off
legend(outLeg);
axis([-2 2 -2 2]);axis square;
title({'Beam Profile Polar Plot (2 Mics)';''})
```

b. simulation_4mic.m

```
clc; clear all;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%% simulation_4mic.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% This is a script to simulate the response of a 4 microphone array to a
%% sinusoidal input incident at varying angle. The results are displayed in
%% the form of a beam profile.

%% defining the system

% 1   2   3   4   component labels
% o   o x o   o   microphones (o) and center of the array (x)

v = 345; %speed of sound in m/s
d = .05; %distance between microphones in m
l = 1; %distance of signal source from array center in m

FS = 100000; %sampling frequency
deltaT = 1/FS; %time step
n = 1000; %number of data points
t = 0:deltaT:n*deltaT;
% f = [300 1000 2000 3000 5000 10000 15000]; %frequencies
f = [300 1000 2000 3000 5000];
deltaTheta = 5;

%iterate over frequencies
for j=1:length(f)
```

```
%% creation of microphone data

%the signal
s = sin(2*pi*f(j)*t);

%iterate over thetas
for theta=0:deltaTheta:360

    %receiving the signal in a microphone array
    x = l*cosd(theta);
    y = l*sind(theta);

    %calculate the delay of microphone 1
    h1 = sqrt((x+3*d/2)^2 + y^2);
    delay1 = h1/v;
    index_delay1 = round(delay1/deltaT);

    %calculate the delay of microphone 2
    h2 = sqrt((x+d/2)^2 + y^2);
    delay2 = h2/v;
    index_delay2 = round(delay2/deltaT);

    %calculate the delay of microphone 3
    h3 = sqrt((x-d/2)^2 + y^2);
    delay3 = h3/v;
    index_delay3 = round(delay3/deltaT);

    %calculate the delay of microphone 4
    h4 = sqrt((x-3*d/2)^2 + y^2);
    delay4 = h4/v;
    index_delay4 = round(delay4/deltaT);

    for i=1:n+1
        %signal from microphone 1
        if i <= index_delay1
            m1(i) = 0;
        else
            m1(i) = s(i-index_delay1);
        end

        %signal from microphone 2
        if i <= index_delay2
            m2(i) = 0;
        else
            m2(i) = s(i-index_delay2);
        end

        %signal from microphone 3
        if i <= index_delay3
            m3(i) = 0;
        else
            m3(i) = s(i-index_delay3);
        end

        %signal from microphone 4
```

```
            if i <= index_delay4
                m4(i) = 0;
            else
                m4(i) = s(i-index_delay4);
            end
        end

        %% applying directionality

        delay = d/v;
        index_delay = round(delay/deltaT);

        %delay signals from microphones 1,2,3
        for i=1:n+1
            if i > index_delay*3
                m1Delay(i) = m1(i-index_delay*3);
            else
                m1Delay(i) = 0;
            end

            if i > index_delay*2
                m2Delay(i) = m2(i-index_delay*2);
            else
                m2Delay(i) = 0;
            end

            if i > index_delay*1
                m3Delay(i) = m3(i-index_delay*1);
            else
                m3Delay(i) = 0;
            end
        end

        %shift signals to only look at the steady state
        ssShift = max([index_delay1 index_delay2 index_delay3
index_delay4])+index_delay;
        m1DelaySS = m1Delay(ssShift:length(m1Delay));
        m2DelaySS = m2Delay(ssShift:length(m2Delay));
        m3DelaySS = m3Delay(ssShift:length(m3Delay));
        m4SS = m4(ssShift:length(m4));
        tSS = t(1:length(t)-ssShift+1);

        %sum and find the amplitude of the resultant signal
        output = m1DelaySS + m2DelaySS + m3DelaySS + m4SS;
        outputMag(floor(theta/deltaTheta)+1) = max(output);

    end
    dirnum(j) = max(outputMag)/mean(outputMag);
    directionalMag(j,:) = outputMag;

end
```

```
%% plot the beam profile

figure;
outLeg = '';
color = ['k' 'r' 'c' 'm' 'b'];
for i=1:length(f)
    theta = (0:deltaTheta:360).*pi/180;
    curr_color = color(mod(i,5)+1);
    polar(theta,directionalMag(i,:),curr_color);hold on;
    outLeg = strvcat(outLeg,sprintf('%.1f Hz/Directionality = %.1f',
f(i),max(dirnum(i))));
end
hold off
legend(outLeg);
axis([-4 4 -4 4]);axis square;
title({'Beam Profile Polar Plot (4 Mics)';''})
```

D. Data Acquisition

a. stepper_daq.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%% stepper_daq.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% stepper.m inititalizes and executes a data acquisition routine. At equal
%% intervals around a full 360 degrees, the script sequentially plays tones
%% at the specified frequency and logs the data collected by a four
%% microphone array. The script then iterates to the next angle and
%% repeats, all the while logging data to a final data matrix.

%% Format of the resultant data matrix with N angles and M frequencies

%% Sampling Rate    n   numfreq         numangle
%% Frequency Array
%% Amplitude Array
%% Angle Array
%% Angle 1              Freq 1          Time
%%                                      Channel 1
%%                                      Channel 2
%%                                      Channel 3
%%                                      Channel 4
%%                      Freq 2->(M-1)
%%                      Freq M          Time
%%                                      Channel 1
%%                                      Channel 2
%%                                      Channel 3
%%                                      Channel 4
%% Angle 2->(N-1)
%% Angle N              Freq 1          Time
%%                                      Channel 1
```

```
%%                                      Channel 2
%%                                      Channel 3
%%                                      Channel 4
%%                      Freq 2->(M-1)
%%                      Freq M          Time
%%                                      Channel 1
%%                                      Channel 2
%%                                      Channel 3
%%                                      Channel 4

Fs = 100000;
n = 10000;
freqs = [300 600 1500 2000 3250 5000]; %frequency array in Hz
amps = [350 150 80 93 65 65];
numfreq = length(freqs);
tics = 1; % must go into 96 evenly for exact revolutions
numangle = 96/tics;

data(1,1) = Fs;
data(1,2) = n;
data(1,3) = numfreq;
data(1,4) = numangle;
data(2,1:numfreq) = freqs;
data(3,1:numfreq) = amps;

for i=1:numangle
    if (180-360*(i-1)/numangle) >= 0
        data(4,i) = 180-360*(i-1)/numangle;
    else
        data(4,i) = 540-360*(i-1)/numangle;
    end
end

T = .2; %time each frequency is to be played in seconds
deltaT = 1; %time between each tone in seconds

% set up daq system with 4 channels of analog input for receiving microphone
% signals and 2 channels of digital output for controlling the stepper
% motor
ai = analoginput('nidaq','Dev1')
ai.SampleRate = Fs;
ai.InputType = 'SingleEnded'
ai.SamplesPerTrigger = n;
ai.TriggerType = 'Immediate';
c1 = addchannel(ai,1,'chan1');
c1.InputRange = [-5 5];c1.SensorRange = [-5 5];c1.UnitsRange = [-5 5];
c2 = addchannel(ai,2,'chan2');
c2.InputRange = [-5 5];c2.SensorRange = [-5 5];c2.UnitsRange = [-5 5];
c3 = addchannel(ai,3,'chan3');
c3.InputRange = [-5 5];c3.SensorRange = [-5 5];c3.UnitsRange = [-5 5];
c4 = addchannel(ai,4,'chan4');
c4.InputRange = [-5 5];c4.SensorRange = [-5 5];c4.UnitsRange = [-5 5];
%2 channels of digital output
dio = digitalio('nidaq','Dev1');
addline(dio, 1:2, 'out'); %dio1 is clock and dio2 is direction
```

```
%start from silence and ready the stepper motor clock signal
setamp(0);
dir=1;
putvalue(dio, [0 dir]);


%step through the angles
for j=1:(96/tics)

    %step through the frequencies
    for i=1:length(freqs)
        setfreq(freqs(i));
        setamp(amps(i));
        pause(T);
        start(ai);
        pause(T);
        setamp(0);
        [d,t]=getdata(ai);
        data(5*i+numfreq*5*(j-1),1:n) = t';
        data(((5*i+1+numfreq*5*(j-1)):(5*i)+4+numfreq*5*(j-1)),1:n) = d';
        pause(deltaT);
    end

    %preserve david's ears
    setamp(0);

    %turn to next angle
    for k=1:tics
        putvalue(dio, [1 dir]);
        putvalue(dio, [0 dir]);
        pause(1/10);
    end
    pause(1)
end
```

b. stepper_adj.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% stepper_adj.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% stepper_adj.m allows for manual adjustment of the step motor position

dio = digitalio('nidaq','Dev1');
addline(dio, 1:2, 'out'); %dio1 is clock and dio2 is direction
dir = 1;%0 is CCW, 1 is CW
putvalue(dio, [0 dir]);

revolution = 96; %% number of tics per revolution

%upper bound of loop is the number of tics that will be executed in the
```

```
%specified direction
for k=1:14
    putvalue(dio, [1 dir]);
    putvalue(dio, [0 dir]);
    pause(.2);
end
```

c. setfreq.m

```
function output = setfreq(freq)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% setfreq.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%% Adapated from Carr Everbach's setfreq.m %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% The parameter freq is the desired amplitude of the signal in hertz.
%% setfreq.m opens HP 3314A function generator at GPIB address 7 and
%% changes the current frequency to frew.

global fungen

if isempty(instrfind)
    fungen = visa('agilent','GPIB0::7::INSTR'); % for Agilent USB-GPIB
end
if fungen.status == 'closed'
    fopen(fungen);
end
freqstr = ['FR',int2str(freq),'HZ'];
fprintf(fungen,'%s\n',freqstr);
fclose(fungen)
```

d. setamp.m

```
function output = setamp(amp)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% setamp.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%% Adapated from Carr Everbach's setamp.m %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% The parameter amp is the desired amplitude of the signal in volts.
%% setamp.m opens HP 3314A function generator at GPIB address 7 and
%% changes the current amplitude to amp.

global fungen
if isempty(instrfind)
    fungen = visa('agilent','GPIB0::7::INSTR'); % for Agilent USB-GPIB
end
```

```
if fungen.status == 'closed'
    fopen(fungen);
end
ampstr = ['AP',int2str(amp),'MV'];
fprintf(fungen,'%s\n',ampstr);
fclose(fungen)
```

E. Processing Experimental Data

a. process.m

```
function returnval = process(data)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%% process.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% The paramater data is a data matrix resulting from a data acquisition
%% sequence from stepper_daq.m. process.m uses repeated calls of
%% process_pair.m to create an output matrix that is eventually returned.
%% The first row of the output matrix is the first row of the data matrix.
%% The second and third rows of the output matrix are the vectors of test
%% frequencies and angles, respectively. The remaining rows of the output
%% matrix correspond to the frequencies specified in the first row where
%% the column entries correspond to the angle specified in the third row.

n=data(1,2);
numfreq = data(1,3);
numangle = data(1,4);

output(1,:) = data(1,:);
output(2,1:numfreq) = data(2,1:numfreq);
output(3,1:numangle) =  data(4,1:numangle);

%4th order butterworth filter to try and get rid of electrical noise
[B,A] = butter(4,.004,'high');

%filter and process data at each angle,frequency pair
for i=1:numfreq*numangle
    i
    currdata = data(5*i-4:5*i,1:n);
    filtereddata(1,:) = currdata(1,:);
    for j=1:4
        filtereddata(j+1,:) = filter(B,A,currdata(j+1,:));
    end
    entry = process_pair([data(1,:) ; filtereddata]);
    output(mod(i-1,numfreq)+4,ceil(i/numfreq)) = entry;
end

returnval = output;
```

b. process_pair.m

```
function returnval = process_pair(data)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% process_pair.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% The paramater data is a 6 row data matrix with the first row being the
%% first row of a matrix created from stepper_daq.m. The second row is the
%% time vector with the next 4 rows being 4 corresponding microphone
%% channels. process_pair.m appropriately delays and sums the 4 microphone
%% channels and returns the result of the findmax algorithm on the
%% calculated sum as its returnval.


v = 345; %speed of sound in m/s
d = .05; %distance between microphones in m

FS = data(1,1); %sampling frequency
deltaT = 1/FS; %time step

%% applying directionality

delay = d/v;
index_delay = round(delay/deltaT);
n=data(1,2);

%delay signals from microphones 1,2,3
for i=1:n
    if i > index_delay*3
        m1 = data(3,:);
        m1Delay(i) = m1(i-index_delay*3);
    else
        m1Delay(i) = 0;
    end

    if i > index_delay*2
        m2 = data(4,:);
        m2Delay(i) = m2(i-index_delay*2);
    else
        m2Delay(i) = 0;
    end

    if i > index_delay*1
        m3 = data(5,:);
        m3Delay(i) = m3(i-index_delay*1);
    else
        m3Delay(i) = 0;
    end
end
```

```
m4 = data(6,:);

%normalize to account of microphone discrepancies
output = m1Delay/findmax(m1Delay) + m2Delay/findmax(m2Delay) +
m3Delay/findmax(m3Delay) + m4/findmax(m4);

%test the entire array
returnval = findmax(output);

%test a single microphone
%returnval = findmax(m4);
```

c. findmax.m

```
function returnmax = findmax(array)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% findmax.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% findmax.m takes an array of data and proceeds to find the limit where
%% 90% of the array data lies between +/- that limit via a binary algorithm

close all;
guess = 5;
upperbound = 10;
lowerbound = 0;

pct = 1; %ensure we get in the loop

threshold = .90;
tolerance = .01;
n = length(array);

while (abs(pct - threshold) > tolerance),

    %count the number of points within the limit
    inside = 0;
    for i=1:n,
        if abs(array(1,i)) < guess,
            inside = inside + 1;
        end
    end

    pct = inside/n;

    %revise boundaries
    if pct > threshold,
        upperbound = guess;
    else
        lowerbound = guess;
```

```
    end
    %make a new guess
    guess = (upperbound-lowerbound)/2+lowerbound;
end

hold on;
plot(linspace(0,.1,n),array), plot([0 .1], [guess guess], 'r',[0 .1], [-guess
-guess], 'r')
hold off;

returnmax = guess;
```

d. plott.m

```
function plott(data)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%% plott.m %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%% David Luong and Mark Piper %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%% e90 Directional Hearing Aid %%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%% Swarthmore College, Spring 2006 %%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% plott.m takes the output matrix from a call to process.m and plots the
%% beam profiles.

close all

freq = data(2,:);
numfreqs = data(1,3);
numangles = data(1,4);

for i=1:numfreqs
    connect = [data(3,1)*pi/180 data(3,numangles)*pi/180 ; data(3+i,1)
data(3+i,numangles)];
    figure; polar(data(3,:)*pi/180,data(3+i,:)); hold on;
polar(connect(1,:),connect(2,:)); hold off;
    title(sprintf('Beam Profile for %0.1f Hz',freq(i)));
    i = i + 1;
end
```